International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 11 | Nov - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

AI-Based Driverless Train with IOT Monitoring

Under the guidance of
Dr. Kaveri
Professor & Head
Department of Electronics and Computer Engineering
Aditya College of Engineering and Technology
Bangalore, India

Gaurav J
Department of Electronics and Computer Engineering
Aditya College of Engineering and Technology
Bangalore, India
gaurav.jnaidu29@gmail.com

Sandeep P
Department of Electronics and Computer Engineering
Aditya College of Engineering and Technology
Bangalore, India
sandeepsandeep54255@gmail.com

Rohith M
Department of Electronics and Computer Engineering
Aditya College of Engineering and Technology
Bangalore, India
gowdarohith640@gmail.com

Venkata Lakshmi
Department of Electronics and Computer Engineering
Aditya College of Engineering and Technology
Bangalore, India
venkatalakshmi634@gmail.com

Abstract- Railway safety still leans heavily on human attention, and spotting an object on the track early enough to react to it remains difficult. This paper describes a miniature driverless train built to explore how low-cost AI and IoT components can help with that problem. A front-mounted camera supplies frames to a YOLO detector running alongside OpenCV, an ultrasonic sensor confirms how far away an object is, and infrared sensors follow marker positions on the track. An ESP32 microcontroller combines these inputs and drives the DC motors through a motor driver, slowing or stopping the train when its path is blocked. Each detection is saved as an image, listed on a Django web dashboard, and sent to the operator as a Telegram message. The model was assembled and tested on a small track as an academic project, and the paper explains its design, its working sequence, and the observations recorded during testing . The work shows that detection, sensing, control, and remote reporting can be integrated on affordable hardware, and it offers a base for further study of autonomous rail systems.

Keywords: Artificial Intelligence, Driverless Train, Internet of Things, YOLO, OpenCV, ESP32, Obstacle Detection, Track Monitoring, Django, Railway Automation.

I. INTRODUCTION

Trains carry millions of passengers and a large share of the world's freight, yet much of their safety still rests on drivers, signals, and people watching the line. An animal, a fallen branch, or a person on the track is hard to notice early and hard to react to quickly. Cheap cameras, small processors, and internet-connected devices now make it practical to build systems that notice such hazards automatically.

Computer vision lets a machine pick out objects in an image, and IoT lets separate devices share what they sense with people who are somewhere else. Put together, they suggest a simple idea: a train that looks ahead, reacts on its own, and reports what happened. That idea is what this project tests on a small scale.

We built a miniature train that uses a camera and a YOLO model, supported by OpenCV, to recognise obstacles and features of the track. An ultrasonic sensor adds a distance reading, and infrared sensors detect marker strips laid along the rails. An ESP32 microcontroller receives these inputs and decides how the DC motors should behave, with a motor driver supplying the power.

On the software side, a Django web application shows the train's status and recent detections, Python scripts handle processing and image storage, and a Telegram bot pushes alerts to the operator's phone. This means the train can be monitored without standing beside it.

The aim of the work is to show how these pieces fit together in one working model and to give students a platform on which each part can be tested and improved later. The rest of the paper reviews related studies, explains the method, compares our approach with conventional monitoring, and lists the benefits and the directions we would follow next.

II. LITERATURE SURVEY

Interest in using vision, sensor networks, and wireless links to watch over railways has grown steadily. Most published studies look at one of these technologies at a time, which gave us a set of building blocks to draw on.

A. Vision-Based Object Detection

Redmon et al. introduced YOLO, which finds and labels objects in a single pass of a neural network instead of running separate proposal and classification stages . Its speed makes it a natural fit for live video, which is why we adopted it for a moving train.

B. Obstacle Detection in Railway Environments

Zhang et al. paired a YOLO-based detector with a risk-assessment step, so that an alarm depends on where an object sits relative to the rails and how dangerous it is, not only on whether it is present . Their work reinforced our decision to treat only objects inside the track region as obstacles.

C. IoT-Based Railway Monitoring

Wei and Song used passive wireless sensors to track the health of railway vehicles while in transit and to pass the readings on for analysis . The study is a useful reference for sending onboard sensor data to a remote viewer.

D. Remote Monitoring and Intelligent Infrastructure

Armijo and Zamora-Sánchez connected the structural monitoring of a railway bridge to an IoT platform and a digital twin, allowing engineers to review conditions remotely . It shows how rail-related data can be gathered in the field and examined elsewhere.

E. Research Gap and Motivation

Each of these studies addresses a single task: detection, in-transit sensing, or infrastructure monitoring. Few bring camera-based detection, distance and marker sensing, motor control, and remote reporting into one working train. Our prototype does this so the interactions between the parts can be seen and measured.

III. METHODOLOGY

The work was carried out in stages: capturing and analysing images, adding sensor readings, controlling the motors, and finally publishing the results to a web dashboard and a messaging app.

A. System Overview

The system has five parts: a front-facing camera, a processing unit that runs the detection software, an ESP32 board with its sensors, a motor driver with DC motors, and a web and messaging layer. The camera and detector decide what lies ahead, the ESP32 decides what the train should do about it, and the web layer tells the operator what happened.

B. Image Acquisition and Processing

The camera looks down the track from the front of the train. Frames are read with OpenCV in Python, resized, and normalised before being passed to the detector. This step keeps the input consistent even when lighting changes slightly between frames.

C. YOLO-Based Obstacle and Track Detection

The detector (YOLO version, model size, weights or training data, input resolution) returns bounding boxes, class labels, and confidence scores. Detections below a confidence threshold are ignored. A box counts as an obstacle only when it overlaps the track region of the frame, which prevents objects beside the rails from stopping the train.

D. Sensor-Based Monitoring

A forward-facing ultrasonic sensor reports the distance to the nearest object. The system slows the train when this distance falls below 30cm and stops it below 20cm. Infrared sensors read reflective markers on the track, which tell the train when it has reached a defined position such as a station stop. These readings back up the camera when an image is unclear.

E. Embedded Control and Train Movement

The ESP32 reads the sensors and receives the vision result over [Wi-Fi / serial / HTTP / MQTT]. Its program has three behaviours: run at normal speed, slow down when an object is detected at medium range, and stop when the object is within the emergency distance. The motor driver converts these decisions into power for the DC motors, and the train moves off again once the path is clear.

F. IoT-Based Monitoring and Notification

Whenever an obstacle is flagged, the frame is stored with a timestamp. The Django application lists recent events, shows the stored images, and displays the current state of the train. At the same moment, a Telegram message carrying the event details goes to the operator.

G. Overall Working Procedure

In a normal run the camera streams frames while the ESP32 polls its sensors. If the track is clear, the train keeps moving. If an obstacle appears, the detector result and the distance reading are compared, the motors slow or stop, an image is saved, and the dashboard and Telegram bot are

updated. When the obstacle is removed, the train resumes its journey.

H. Test Setup and Observations

Track length, lighting conditions, train speed, obstacle types, number of trials, correct detections, missed detections, false alarms, stopping distance, and the delay from detection to motor stop and to Telegram alert.

IV. EXISTING SYSTEM

Conventional railways depend on operators, signalling equipment, and trackside devices to control train movement. These arrangements work well for scheduled operation, but continuous watching of the line and quick identification of unexpected objects remain hard problems.

Cameras and sensors are often installed as separate systems. Because their outputs are not combined, building one clear picture of what is happening around the train takes extra effort, and manual inspection adds time and staffing.

Detection, train control, and infrastructure monitoring are also commonly handled by different tools. When these operate independently, sharing their data is harder, and remote access to events and images depends on whatever infrastructure is available. This gap motivated a design in which all of these functions share one pipeline.

V. PROPOSED SYSTEM

The proposed system joins AI, computer vision, embedded electronics, and IoT monitoring in a single miniature train. It is meant to show how such a vehicle can watch its own path, respond to obstacles, and report what it finds.

A. Camera-Based Detection System

A camera mounted at the front supplies video that OpenCV prepares and a YOLO model analyses, giving the train a view of obstacles and track features ahead.

B. Intelligent Obstacle Monitoring

A camera can be fooled by glare or partial views, so the ultrasonic distance reading is used as a second check. A vision detection supported by a short distance is treated as a confirmed obstacle, while a very short distance alone is enough to trigger an emergency stop.

C. Track Feature Identification

The vision module looks for visible track features, and infrared sensors detect marker strips placed on the miniature track. Using both lets the train recognise where it is on the route.

D. ESP32-Based Control Unit

The ESP32 acts as the central controller. It gathers sensor inputs, applies the stopping and slowing rules, and sends commands to the motor driver.

E. Motor Control Mechanism

The motor driver supplies the DC motors according to the ESP32's signals, which allows forward motion at different speeds and quick stops.

F. Web-Based IoT Monitoring

The Django dashboard displays system status and stored detection images, while Telegram alerts give the operator an additional, immediate channel.

G. Integrated System Operation

Taken together, the camera, detection software, sensors, controller, and motors form one loop: sense, decide, act, and report. Because each part is separate, any of them can be replaced or upgraded without redesigning the rest.

VI. ADVANTAGES

A. Automated Train Movement

The controller starts, slows, and stops the train on its own, reducing the need for constant manual operation.

B. Visual Obstacle Detection

YOLO and OpenCV let the train recognise what is on the track rather than reacting only to distance.

C. Additional Sensor Support

Ultrasonic and infrared sensors provide a second source of information that supports the camera.

D. Remote System Monitoring

The web dashboard lets an operator check the train from any connected device.

E. Event Notifications

Telegram alerts report detections immediately, so no one has to watch the screen continuously.

F. Integrated Technology Platform

AI, vision, embedded control, and IoT work together in one inexpensive model that is easy to demonstrate.

G. Scope for Future Enhancement

The modular layout makes it straightforward to upgrade the detector, add sensors, and extend monitoring.

VII. CONCLUSION

This paper presented a miniature driverless train in which a YOLO and OpenCV vision module, ultrasonic and infrared sensing, and an ESP32 controller work together to detect obstacles and manage movement. A Django dashboard, stored images, and Telegram alerts keep an operator informed from a distance.

The project shows that these technologies can be combined on low-cost hardware and gives us a platform for studying railway automation. As a small-scale model tested in a controlled setting, it is a starting point rather than a finished railway product. Our next steps are testing under varied lighting and track conditions, training the detector on rail-specific data, adding more sensors, and measuring reliability over longer runs.

VIII. REFERENCES

[1] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 779–788.

[2] Z. Zhang, P. Chen, Y. Huang, L. Dai, F. Xu, and H. Hu, "Railway obstacle intrusion warning mechanism integrating YOLO-based detection and risk assessment," Journal of Industrial Information Integration, vol. 38, Art. no. 100571, 2024.

[3] J. Wei and Y. Song, "Passive Wireless Sensor Network-Based In-Transit Health Status Monitoring for Railway Transportation," Journal of Sensors, vol. 2021, Art. no. 3789699, 2021.

[4] A. Armijo and D. Zamora-Sánchez, "Integration of Railway Bridge Structural Health Monitoring into the Internet of Things with a Digital Twin: A Case Study," Sensors, vol. 24, no. 7, Art. no. 2115, 2024.